PyACTS:
A High-Level User Interface to The ACTS Collection

Tony Drummond
Computational Research Division
Lawrence Berkeley National Laboratory

Vicente Galiano
Miguel Hernandez University

Violeta Migallón and José Penadés
University of Alicante

ACTS
Collection

Highlights of the Project

Osni A. Marques (PI) **Tony Drummond (co-PI)**

Goal: The Advanced CompuTational Software Collection (ACTS) makes reliable and efficient software tools more widely used, and more effective in solving the nation's engineering and scientific problems.

Components:

- **Solid Base:** non-commercial and open source tools developed at DOE laboratories and universities.
- **Independent Tool Evaluations and Consultation** provided through acts-support@nersc.gov
- **High Level User Support** problem identification, tool and interface selection, specific tuning parameter configurations, installation, documentation, etc.
- **Training and Dissemination** workshops, lectures, active conference participation (acts.nersc.gov.
- **Collaborations** with HPC centers, computational sciences research centers (national and international level), and software and computer vendors.

Office of Science
U.S. DEPARTMENT OF ENERGY

ACTS Tools — *Available Functionality*

Numerical $Ax = b$ $Az = \lambda z$ $A = U\Sigma V^T$ PDEs ODEs $\vdots$	**Aztec**	Algorithms for the iterative solution of large sparse linear systems.
	Hypre	Library of preconditioners for the solution of PDEs.
	PETSc	Toolkit to support the solution of PDEs.
	OPT++	Object-oriented nonlinear optimization package.
	SUNDIALS	Solvers for the solution of systems of ordinary differential equations, nonlinear algebraic equations, and differential-algebraic equations.
	ScaLAPACK	Library of high performance dense linear algebra routines.
	SuperLU	General-purpose library for the direct solution of large, sparse, nonsymmetric systems of linear equations.
	TAO	Large-scale optimization software, including nonlinear least squares, unconstrained minimization, bound constrained optimization, and general nonlinear optimization.
Code Development	**Global Arrays**	Library for writing parallel programs that use large arrays distributed across processing nodes and that offers a shared-memory view of distributed arrays.
	Overture	Object-Oriented tools for solving problems in complex geometries.
Code Execution	**CUMULVS**	E enables programmers to incorporate fault-tolerance, interactive visualization and computational steering into existing parallel programs
	Globus	Services for the creation of computational Grids and tools with which applications can be developed to access the Grid.
	TAU	T tools for analyzing the performance of C, C++, Fortran and Java programs.
Library Development	**ATLAS**	Tools for the automatic generation of optimized numerical software for modern computer architectures and compilers.

ACTS Collection

Software Reusability

What have we gained? What are the goals?

min[*time_to_first_solution*] (prototype)

min[*time_to_solution*] (production)

- Outlive Complexity
  - Increasingly sophisticated models
  - Model coupling
  - Interdisciplinary

(Software Evolution)

- Sustained Performance
  - Increasingly complex algorithms
  - Increasingly diverse architectures
  - Increasingly demanding applications

(Long-term deliverables)

min[*software-development-cost*]

max[*software_life*] and **max**[*resource_utilization*]

ACTS Collection

User Interfaces

```
CALL BLACS_GET( -1, 0, ICTXT )
 CALL BLACS_GRIDINIT( ICTXT, 'Row-major', NPROW, NPCOL )
:
 CALL BLACS_GRIDINFO(ICTXT, NPROW, NPCOL, MYROW, MYCOL)
:
:
 CALL PDGESV( N, NRHS, A, IA, JA, DESCA, IPIV, B, IB, JB, DESCB,
 $                INFO )
```

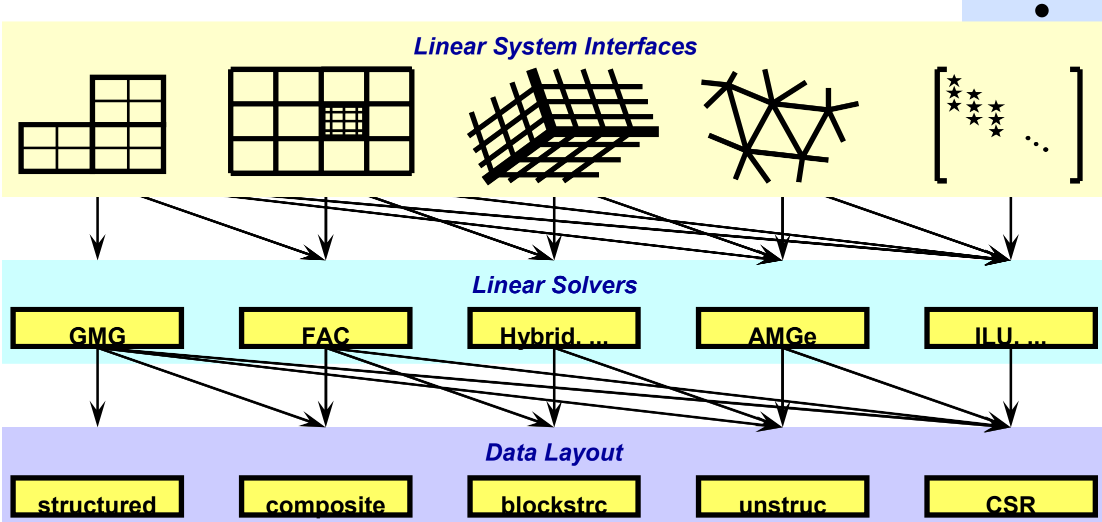
Library Calls

- -ksp_type [cg,gmres,bcgs,tfqmr, …]
- -pc_type [lu,ilu,jacobi,sor,asm, …]

More advanced:

- -ksp_max_it <max_iters>
- -ksp_gmres_restart <restart>
- -pc_asm_overlap <overlap>

Command lines

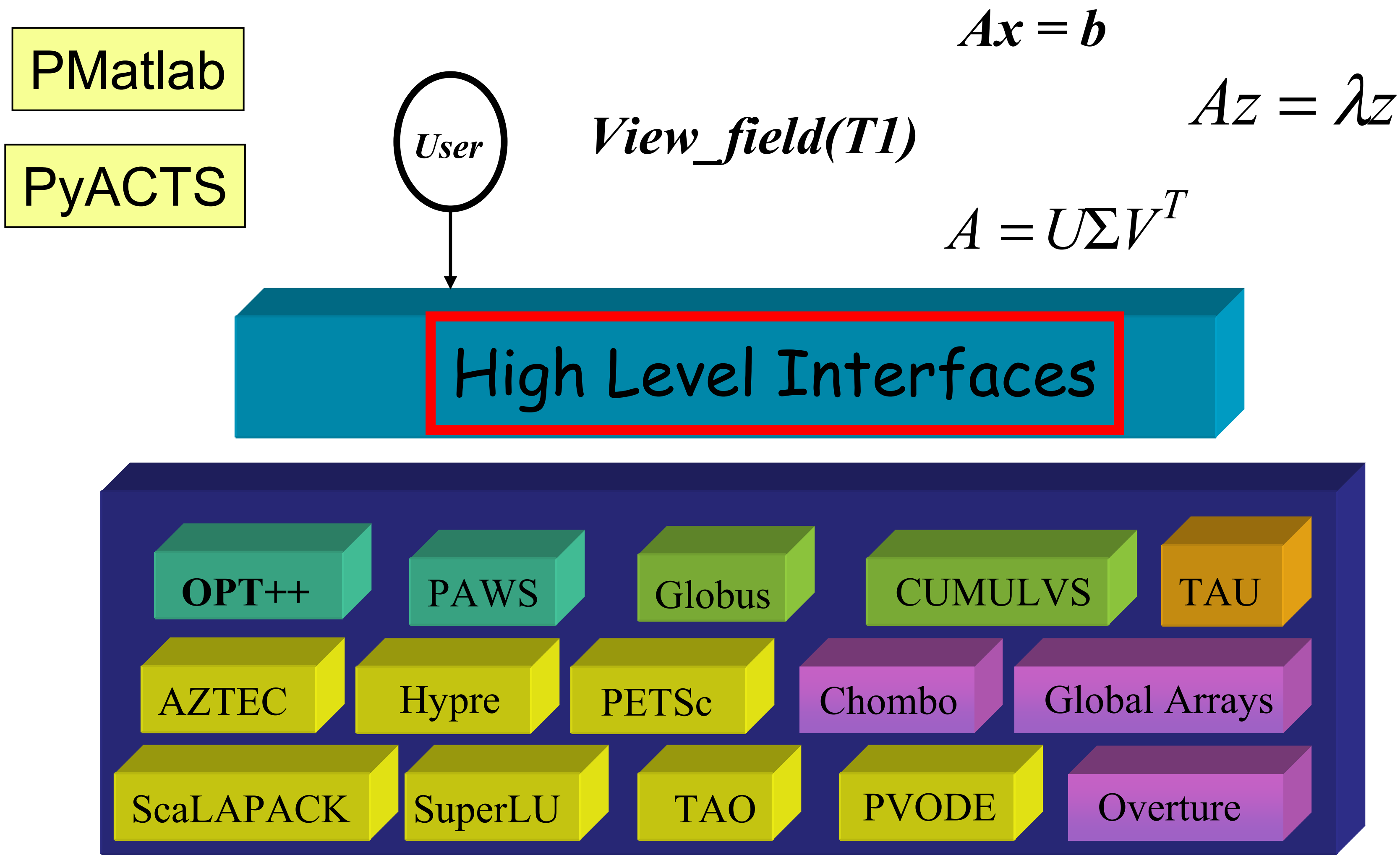


Problem Domain

PyACTS Motivation and Design Consideration

- High-level user friendly interface
- Hides details of parallelism from users
- Teaches users how to use the tools
- Flexible parameter reconfiguration
- Interoperability
- Choice of language: Python

PyACTS — Problem Solving Environments



PyACTS

Motivation and Design Consideration

- Choice of scripting language: Python
- Uses PyMPI and Numeric
- Intended for prototyping and not high-performing production runs.

PyACTS Motivation and Design Consideration

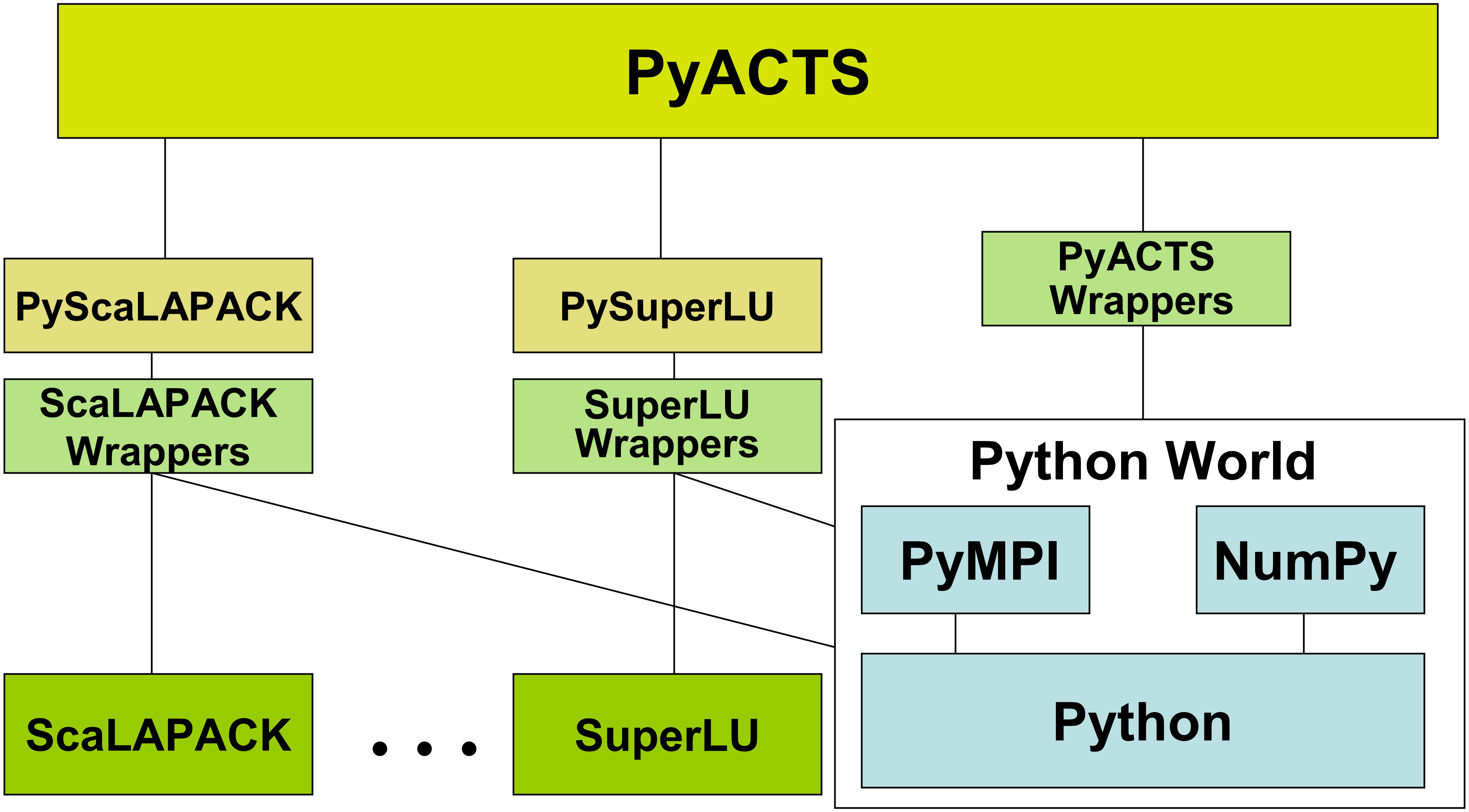
PyClimate (*J. Saenz et al,Univ. Basque Country*)

Support to common tasks during the analysis of climate variability data.

- •Simple IO operations
- •Operations with COARDS-compliant netCDF files
- •Empirical Orthogonal Function (EOF) analysis,
- •Canonical Correlation Analysis (CCA)
- •Singular Value Decomposition (SVD) analysis of coupled datasets
- •Some linear digital filters
- •Kernel based probability-density function estimation and
- •access to DCDFLIB.C library from Python.

PyACTS

A Conceptual View Of PyACTS



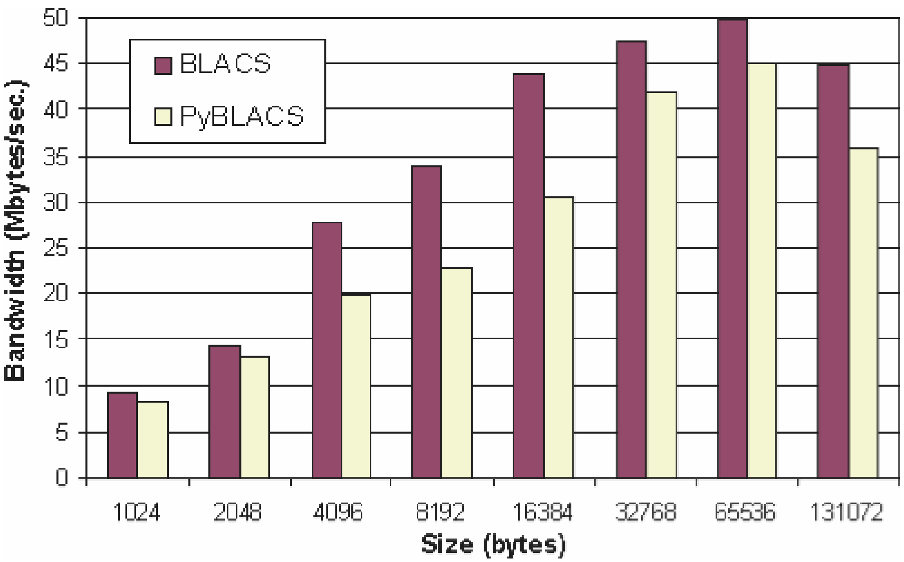
PyACTS

PyACTS Services

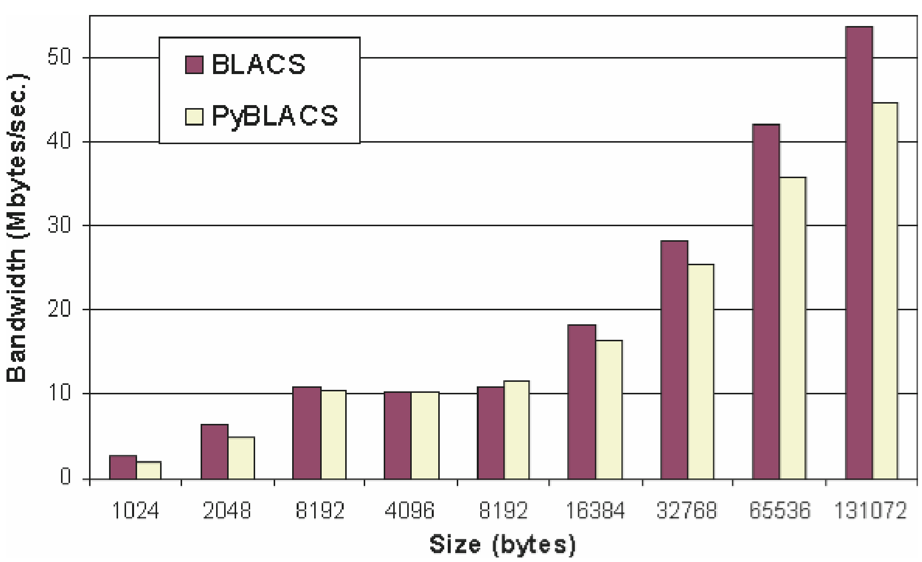
- **BASIC Services:** Creation and modification of different data objects and parallel environment specifications (matrices, data layouts, ctx,)
- **I/O Services :** Parallel read/write. Currently supported ASCII and NetCDF.
- **Verification and Validation:** Predicates and parameter type checking.
- **Data Conversion.** Interoperable objects between libraries.

PyScaLAPACK

BLACS vs PyBLACS



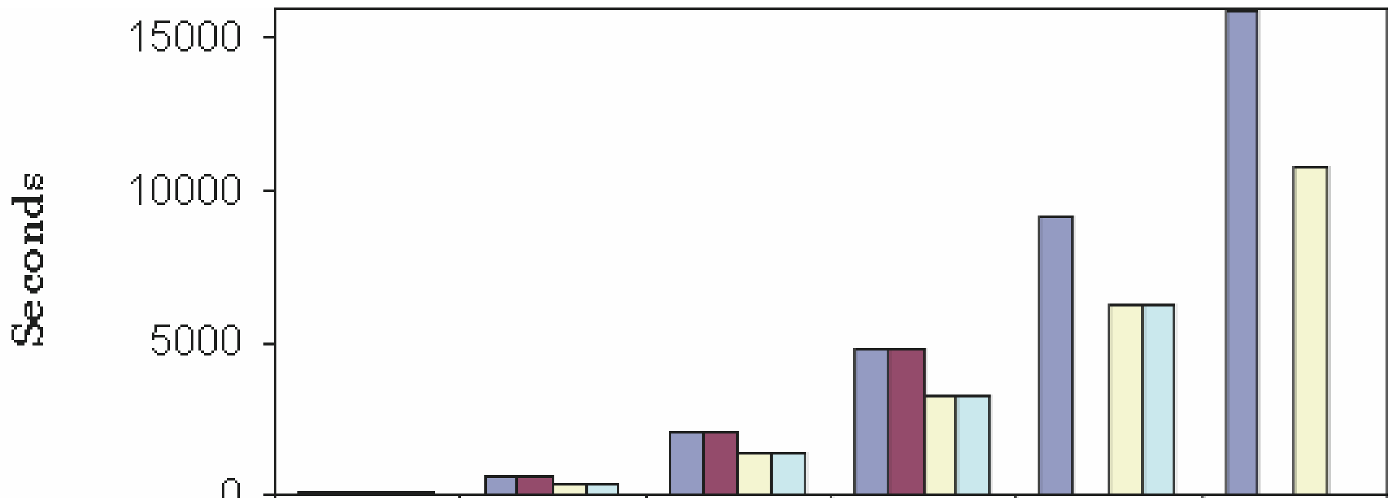
Linux Cluster (2Ghz)



IBM SP - PWR 3

PyScaLAPACK

Example of PBLAS: pdgemm



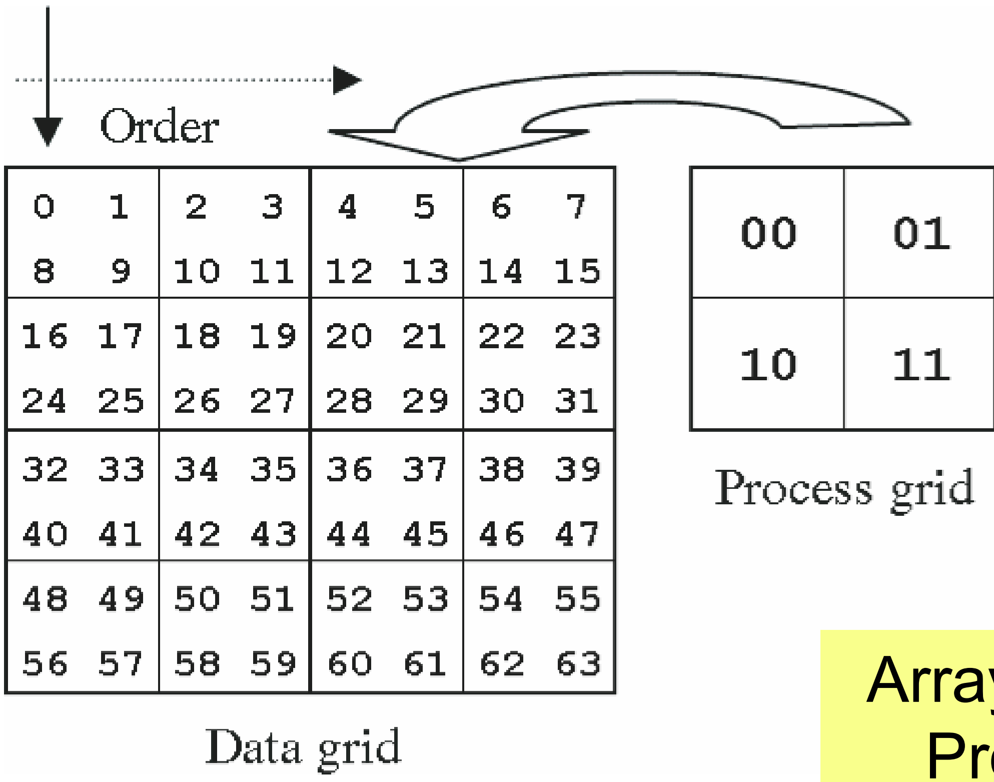
	2000	4000	6000	8000	10000	12000
PyPBLAS 2x2	76,0	608,3	2038,7	4812,6	9149,1	15890,7
PBLAS 2x2	76,0	610,2	2043,3	4805,9		
PyPBLAS 3x2	51,3	403,1	1360,0	3236,3	6319,4	10826,5
PBLAS 3x2	51,0	404,0	1367,2	3241,1	6310,4	

Matrix size

Office of Science
U.S. DEPARTMENT OF ENERGY
ACTS COLLECTION

PyScaLAPACK

PyBLACS Example Operation

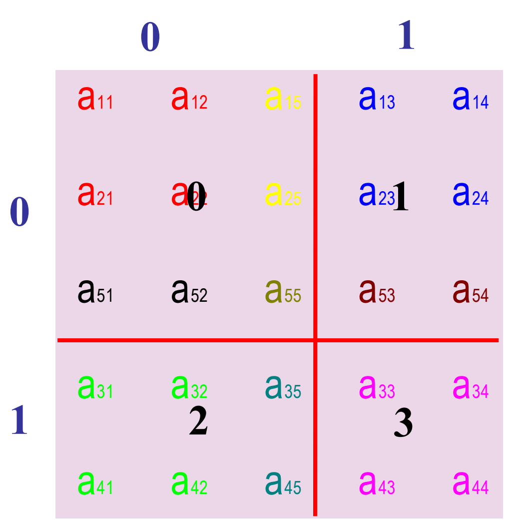


Array Distribution and Processor Layout

PyScaLAPACK

BLACS Equivalent

$$\begin{bmatrix} 1.1 & 1.2 & 1.3 & 1.4 & 1.5 \\ -2.1 & 2.2 & 2.3 & 2.4 & 2.5 \\ -3.1 & -3.2 & 3.3 & 3.4 & 3.5 \\ -4.1 & -4.2 & -4.3 & 4.4 & 4.5 \\ -5.1 & -5.2 & -5.3 & -5.4 & 5.5 \end{bmatrix}$$




```
CALL BLACS_GRIDINFO( ICTXT, NPROW, NPCOL, MYROW, MYCOL )

IF   ( MYROW.EQ.0 .AND. MYCOL.EQ.0 ) THEN
     A(1) = 1.1; A(2) = -2.1; A(3) = -5.1;
     A(1+LDA) = 1.2; A(2+LDA) = 2.2; A(3+LDA) = -5.2;
     A(1+2*LDA) = 1.5; A(2+3*LDA) = 2.5; A(3+4*LDA) = -5.5;
ELSE IF ( MYROW.EQ.0 .AND. MYCOL.EQ.1 ) THEN
     A(1) = 1.3; A(2) = 2.3; A(3) = -5.3;
     A(1+LDA) = 1.4; A(2+LDA) = 2.4; A(3+LDA) = -5.4;
ELSE IF ( MYROW.EQ.1 .AND. MYCOL.EQ.0 ) THEN
     A(1) = -3.1; A(2) = -4.1;
     A(1+LDA) = -3.2; A(2+LDA) = -4.2;
     A(1+2*LDA) = 3.5; A(2+3*LDA) = 4.5;
ELSE IF ( MYROW.EQ.1 .AND. MYCOL.EQ.1 ) THEN
     A(1) = 3.3; A(2) = -4.3;
     A(1+LDA) = 3.4; A(2+LDA) = 4.4;
END IF

CALL PDGESVD( JOBU, JOBVT, M, N, A, IA, JA, DESCA, S, U, IU,
        JU, DESCU, VT, IVT, JVT, DESCVT, WORK, LWORK,
        INFO )
```

PyScaLAPACK

Example of PBLAS: pvgemm

```
PvGEMM( TRANSA, TRANSB, M, N, K, ALPHA,
        A, IA, JA, DESCA,
        B, IB, JB, DESCB,
        BETA, C, IC, JC, DESCC )
```

- User needs to know about the parallel environment (data layout)
- User needs to initialize the process grid (BLACS)
- User needs to distribute data arrays
- Know details about the BLAS 3 call

PyScaLAPACK

Example of PBLAS: pvgemm

```
> from PyACTS import *
> import PyACTS.PyPBLAS as PyPBLAS
> import time
> n=500
> ACTS_lib=1 # ScaLAPACK library
> PyACTS.gridinit() # grid initialization
> alpha=Scal2PyACTS(2,ACTS_lib) # convert scalar
                                # to PyACTS scalar
> beta=Scal2PyACTS(3,ACTS_lib)
> a=Rand2PyACTS(n,n,ACTS_lib) # generate a random
                              # PyACTS array
> b=Rand2PyACTS(n,n,ACTS_lib)
> c=Rand2PyACTS(n,n,ACTS_lib)
> c=PyPBLAS.pvgemm(alpha,a,b,beta,c) # call level 3
                                     # PBLAS routine
> PyACTS.gridexit()
```

PyScaLAPACK

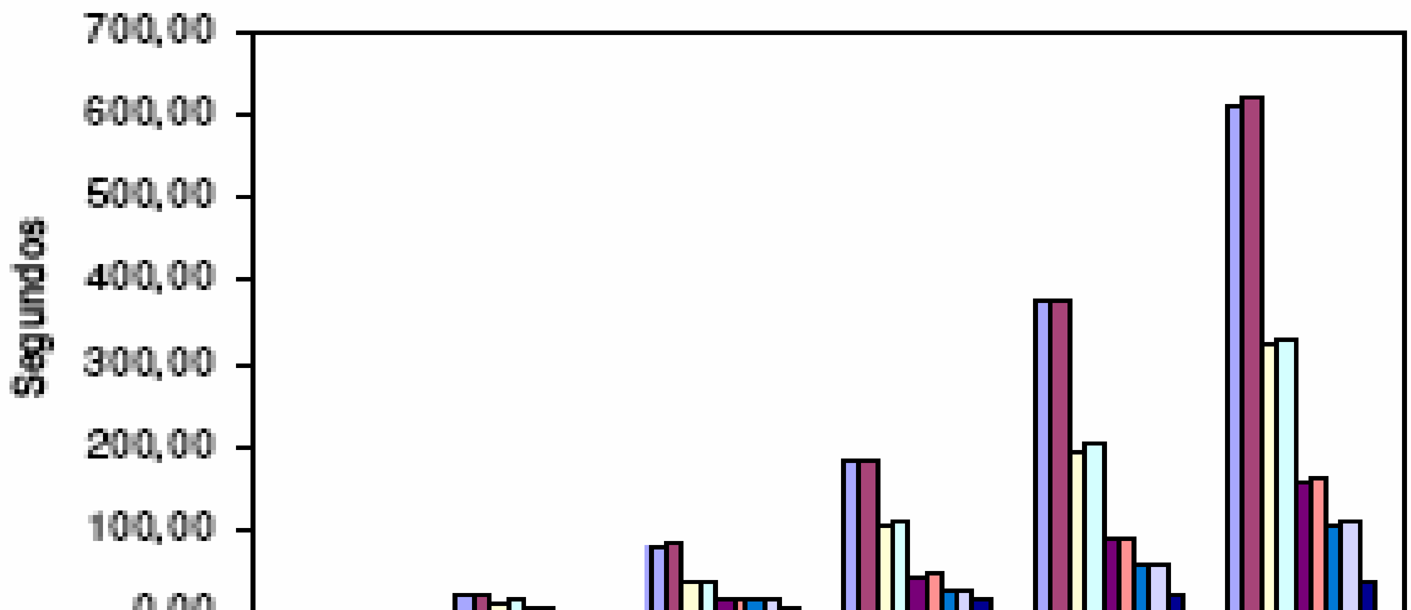
PyBLACS operation

```
PyACTS Array Properties in [ 0 , 0 ]
       lib= 1; desc= [1 0 8 8 2 2 0 0 4]
       data= [ 0  8 32 40  1  9 33 41
               4 12 36 44  5 13 37 45]
PyACTS Array Properties in [ 1 , 0 ]
       lib= 1;desc= [1 0 8 8 2 2 0 0 4]
       data= [16 24 48 56 17 25 49 57
              20 28 52 60 21 29 53 61]
PyACTS Array Properties in [ 1 , 1 ]
       lib= 1; desc= [1 0 8 8 2 2 0 0 4]
       data= [18 26 50 58 19 27 51 59
              22 30 54 62 23 31 55 63]
PyACTS Array Properties in [ 0 , 1 ]
       lib= 1; desc= [1 0 8 8 2 2 0 0 4]
       data= [ 2 10 34 42  3 11 35 43
               6 14 38 46  7 15 39 47]
```

Output from code

PyScaLAPACK

Example of *pvgesvd*



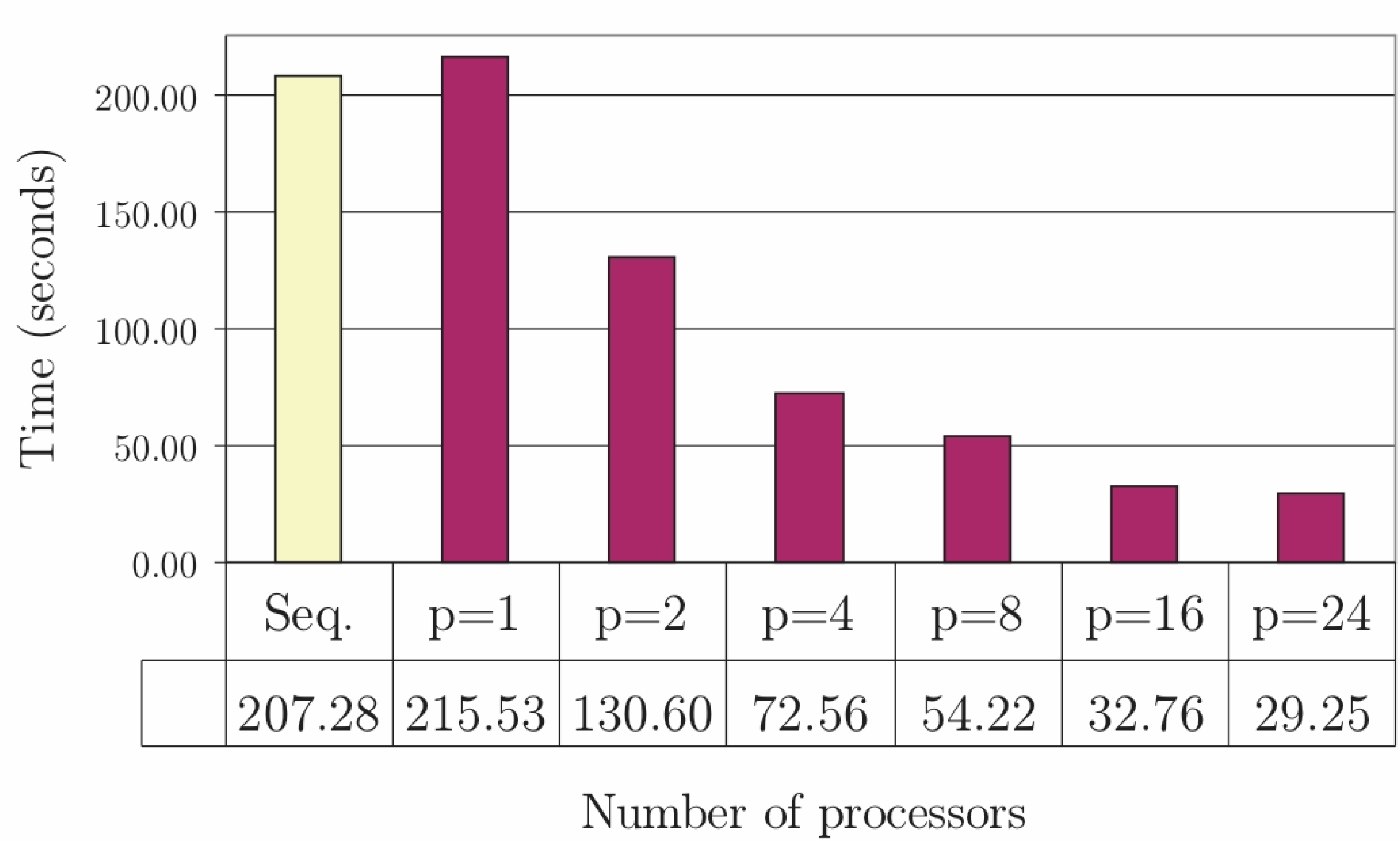
	500,00	1000,00	1500,00	2000,00	2500,00	3000,00
FORTRAN 1X1	2,27	20,05	79,06	184,42	373,37	612,16
PYTHON 1X1	2,42	20,49	80,79	186,65	377,11	616,25
FORTRAN 2X1	1,98	11,69	36,60	102,33	194,60	325,91
PYTHON 2X1	1,95	13,43	40,65	109,44	203,58	328,24
FORTRAN 2X2	1,39	6,86	17,91	44,56	89,53	156,17
PYTHON 2X2	1,46	6,53	17,77	46,64	90,64	164,33
FORTRAN 3X2	1,34	4,77	13,76	28,77	57,48	106,95
PYTHON 3X2	1,34	5,00	13,97	29,23	59,73	111,67
FORTRAN 4X4	1,29	3,27	6,99	14,12	23,20	36,36
PYTHON 4X4	0,21	0,45	0,84	1,40	2,25	3,62

Office of Science
U.S. DEPARTMENT OF ENERGY
ACTS COLLECTION
BERKELEY LAB

PySCaLAPACK

Example from pyClimate

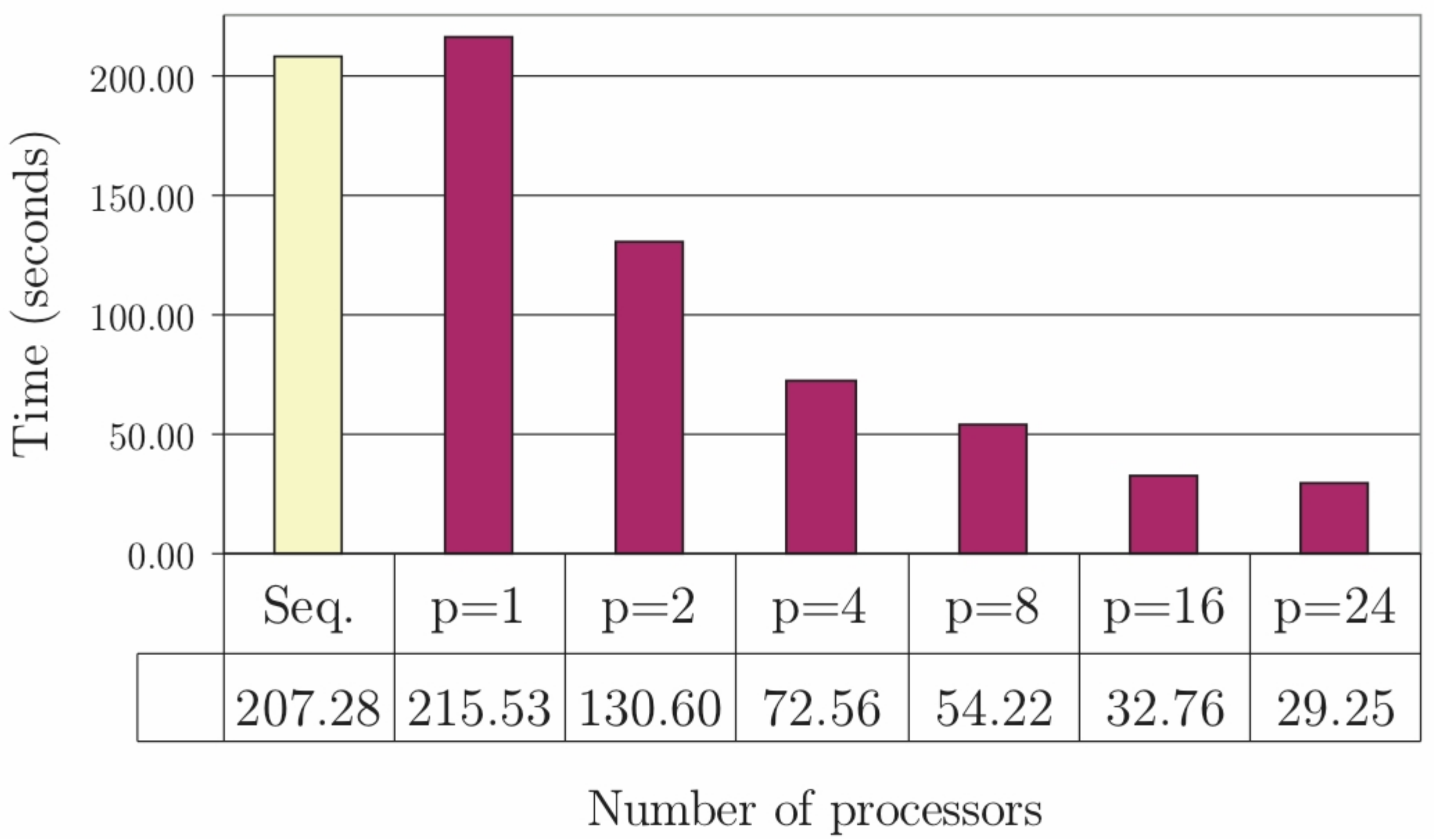
Empirical Orthogonal Function (Day calc)



PyScaLAPACK

Example from pyClimate

Empirical Orthogonal Function (Month calc)



Related Work

- PyTrilinos
- SUNDIALS has a Python interface
- PETSc has a Python interface
- TAU has a profiling interface to Python

Future Work

- Work on release and documentation
- Include other tools and data formats
- Scriber function $\Rightarrow$ high performance codes in C, C++ and Fortran flavors.